(19) JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: 05233690 A

(43) Date of publication of application: 10.09.93

(51) Int. Cl G06F 15/38
G06F 9/44

(21) Application number: 04034864

(22) Date of filing: 21.02.92

(71) Applicant: FUJITSU LTD

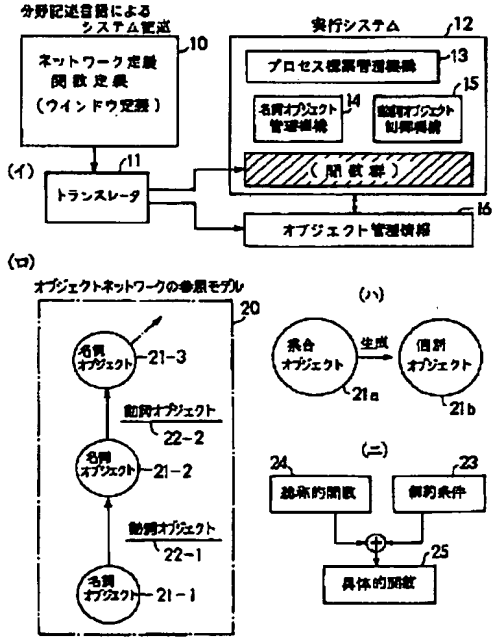
(72) Inventor: ENOMOTO HAJIME
KAMOSHITA MINORU

(54) LANGUAGE PROCESSING SYSTEM BY OBJECT NETWORK

(57) Abstract:

PURPOSE: To easily construct the system of a certain specific applied field, with regard to the language processing system by an object network in a com puter system for managing and processing data of an operation object and an operating means to those data as objects.

CONSTITUTION: The system is constituted so that an object network 20 in which an object is classified roughly into a noun object 21 and a verb object 22, and the noun object 21 and the verb object 22 are expressed as a node and a branch, respectively becomes a reference model, and in this reference model, when the contents of a function as its verb object 22-1 are operated on a noun object 21-1 existing in a certain node, a noun object 21-2 being a certain target object is obtained in the direction of the branch corresponding to its verb object name.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平5－233690

(43)公開日　平成５年(1993)９月10日

(51)Int.Cl.[5]	識別記号	庁内整理番号	Ｆ Ｉ	技術表示箇所
Ｇ０６Ｆ　15/38		Ｍ　9194－5Ｌ		
9/44	３３０	Ｚ　9193－5Ｂ		

審査請求　未請求　請求項の数12(全 23 頁)

(21)出願番号　特願平4－34864

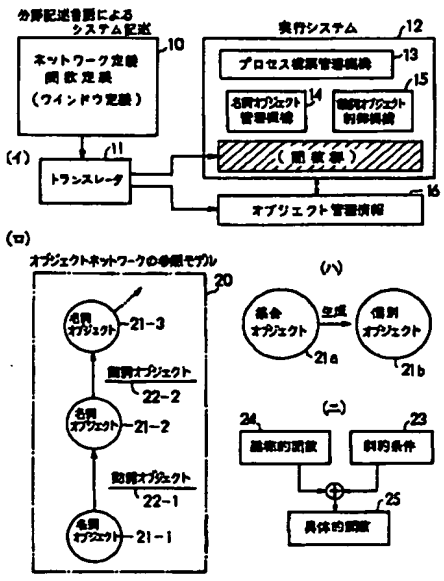
(22)出願日　平成４年(1992)２月21日

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中1015番地
(72)発明者　榎本　肇
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内
(72)発明者　鴨志田　稔
東京都新宿区百人町１－13－２
(74)代理人　弁理士　小笠原　吉義　(外２名)

(54)【発明の名称】　オブジェクトネットワークによる言語処理システム

(57)【要約】
【目的】操作対象のデータおよびそれらのデータに対する操作手段をオブジェクトとして管理し処理する計算機システムにおけるオブジェクトネットワークによる言語処理システムに関し，ある特定の応用分野のシステムを容易に構築できるようにすることを目的とする。
【構成】オブジェクトを名詞オブジェクト21と動詞オブジェクト22とに大分類し，名詞オブジェクト21をノード，動詞オブジェクト22をブランチとして表現したオブジェクトネットワーク20を，参照モデルとし，この参照モデルにおいて，あるノードに存在する名詞オブジェクト21-1に動詞オブジェクト22-1としての関数の内容を作動させると，その動詞オブジェクト名に対応するブランチの方向にある目的対象となっている名詞オブジェクト21-2が得られるように構成する。

本発明の原理説明図



1

【特許請求の範囲】

【請求項1】　操作対象のデータおよびそれらのデータに対する操作手段をオブジェクトとして管理し処理する計算機システムにおいて，前記オブジェクトを名詞オブジェクト(21)と動詞オブジェクト(22)とに大分類し，名詞オブジェクトをノード，動詞オブジェクトをブランチとして表現したオブジェクトネットワーク(20)を，参照モデルとし，この参照モデルにおいて，あるノードに存在する名詞オブジェクトに動詞オブジェクトとしての関数の内容を作動させると，その動詞オブジェクト名に対応するブランチの方向にある目的対象となっている名詞オブジェクトが得られるようにしたことを特徴としたオブジェクトネットワークによる言語処理システム。

【請求項2】　請求項1記載のオブジェクトネットワークによる言語処理システムにおいて，前記名詞オブジェクト(21)については，普通名詞に対応する集合オブジェクト(21a)と，固有名詞に対応する個別オブジェクト(21b)とを有し，それらを名前づけまたは参照指示機能によって区別する名詞オブジェクト管理機構(14)を持つことを特徴とするオブジェクトネットワークによる言語処理システム。

【請求項3】　請求項1または請求項2記載のオブジェクトネットワークによる言語処理システムにおいて，名詞オブジェクトを修飾する形容詞としての制約条件を，名詞オブジェクトに付加し，オブジェクトとして管理する修飾管理機構を持ち，この修飾管理機構は，具体化された個別オブジェクトが属する名詞オブジェクトの持つ性質とそれを修飾する制約条件に対する充足性を判定する手段を持つことを特徴とするオブジェクトネットワークによる言語処理システム。

【請求項4】　請求項1記載のオブジェクトネットワークによる言語処理システムにおいて，前記動詞オブジェクトとして，あるオブジェクトにデータなどの制約条件と併合した演算を施し，目的対象オブジェクトとして名詞オブジェクトを具体化する演算動詞と，集合オブジェクトを構成する個別オブジェクト群に作動して，新しい関係を持った名詞オブジェクトを生成する関係付加動詞と，目的名詞オブジェクトに，制約条件の性質を規定する前置詞と併合する名詞オブジェクトによって具体的制約条件を与え，制約を満足するように目的名詞オブジェクトに対し演算を行う制約演算動詞の少なくとも3種類の機能が存在し，これらが文単位で判定処理されるように構成されたことを特徴とするオブジェクトネットワークによる言語処理システム。

【請求項5】　請求項1記載のオブジェクトネットワークによる言語処理システムにおいて，前記動詞オブジェクトについて，目的対象としての名詞オブジェクトに対し，実際に実行処理が可能な具体的関数(25)と，制約条件(23)が付けられることによって具体的関数に変換される総称的関数(24)の種別を持ち，総称的関数から具体的

2

関数への変換制御を行う動詞オブジェクト制御機構(15)を持つことを特徴とするオブジェクトネットワークによる言語処理システム。

【請求項6】　請求項5記載のオブジェクトネットワークによる言語処理システムにおいて，前記動詞オブジェクト制御機構は，総称的関数に対応する動詞オブジェクトに対する制約条件として，現在処理の実行が終了した名詞オブジェクト名を状態名として制約条件化する機能を持ち，その制約条件をもとに実行可能な具体的関数を得るように構成されていることを特徴とするオブジェクトネットワークによる言語処理システム。

【請求項7】　請求項5記載のオブジェクトネットワークによる言語処理システムにおいて，動詞オブジェクトを関数として作動させるに当って，作動開始前，作動中，終了後の制約条件を付加し，各時点でこれらの妥当性を検定する手段を持つとともに，制約条件が総称的条件で与えられた総称的関数に対し，具体的制約条件を規定することにより具体的関数に変換する過程において逐次的に妥当性判定と変換のための変換手続の生成を行う関数実行管理機構を持つことを特徴とするオブジェクトネットワークによる言語処理システム。

【請求項8】　請求項1記載のオブジェクトネットワークによる言語処理システムにおいて，参照モデルとしての構造を示すオブジェクトネットワーク中の普通名詞に対応する集合オブジェクト状態から，特定の具体的関数の実行によって個別オブジェクトを生成し，それらの要素間関係を表現することにより要素ネットワークを構成するとともに，その要素ネットワークでは，個別オブジェクトについての実行要求によって実際に処理を行うように，オブジェクトネットワークを生成管理するプロセス構築管理機構(13)を持つことを特徴とするオブジェクトネットワークによる言語処理システム。

【請求項9】　請求項8記載のオブジェクトネットワークによる言語処理システムにおいて，前記プロセス管理機構は，処理順序に関する制約条件を，結合されているオブジェクトネットワーク中の動詞オブジェクトの方向性によって規定すると同時に，次にそのオブジェクトが実行可能な状態であることを示す可能性トークンの存在する動詞オブジェクトが並行処理可能であることを判定し，この動詞オブジェクトに対する作動要求を受付けて具体的に実行を行わせるようなプロセスの並行処理のための制御を行うように構成されていることを特徴とするオブジェクトネットワークによる言語処理システム。

【請求項10】　請求項8記載のオブジェクトネットワークによる言語処理システムにおいて，構造を規定する部分を持つオブジェクトネットワークでの動詞オブジェクトが作動して対応する要素ネットワークが作られ，そこに要素オブジェクトに関するデータが制約条件として付加され，このプロセスの実行は，動詞オブジェクトについてのシステム内外からの要求と，処理順序に関する

制約のもとに，前記プロセス構築管理機構の指示制御のもとに並列実行されるように構成されていることを特徴とするオブジェクトネットワークによる言語処理システム。

【請求項１１】　請求項８記載のオブジェクトネットワークによる言語処理システムにおいて，前記プロセス構築管理機構は，参照モデルとなる構造ネットワークに基づいて，新しく構造ネットワークを生成し，参照モデルとしてのオブジェクトネットワークの制約条件を，他システムの要求に従って変更改定し，新しく生成されたオブジェクトネットワークの制約条件を規定し，その結果完成した構造ネットワークが行うべき処理順序に基づいて要素ネットワークへの要求を他システムから受付け可能とすることにより，制御権限を委託し，その結果処理が完了したことの受付けをし，構造ネットワークを消滅させ，この動作が同時に要素ネットワークの完成となるようにされ，処理結果として他システムへ表示する制御手段を持つことを特徴とするオブジェクトネットワークによる言語処理システム。

【請求項１２】　請求項８記載のオブジェクトネットワークによる言語処理システムにおいて，プロセス構築管理機構(13)から構造ネットワークの生成処理要求，その結果の応答，また構造ネットワークから要素ネットワークの生成処理要求および応答というように，要求と応答とを対として，下層ネットワークへの要求，上層ネットワークへの応答を繰り返すサービスを，各要求に対応する関数の実行により行う関数実行管理機構を持ち，この関数実行管理機構は，総称的関数から具体的関数への変換，実行上の妥当性検定の指示制御を行い，その結果不備であれば新しい要求と応答との対を，逆に上層ネットワークへ向けて発行し，完備させ，完備していれば実行し，上層ネットワークへの応答の処理を行うように構成されていることを特徴とするオブジェクトネットワークによる言語処理システム。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は，操作対象のデータおよびそれらのデータに対する操作手段をオブジェクトとして管理し処理する計算機システムであって，ある特定の応用分野のシステムを容易に構築できるようにしたオブジェクトネットワークによる言語処理システムに関する。

【０００２】

【従来の技術】コンピュータの急激な進歩に伴い，システム全体に占めるソフトウェアのコストの割合が増大してきている。そして，システムが巨大になるにつれ，ソフトウェア開発者の労力は指数関数的に増え，ソフトウェア危機と呼ばれる状態が続いている。従来の多くのソフトウェア開発では，開発対象のシステムを，その処理の手続きを記述することにより表現するという，プログラミング（手続き）中心の開発がなされており，これがソフトウェア危機の解決を難しくする原因になっていた。

【０００３】これに対し，手続きを中心に考えるのではなく，処理や操作の対象を中心に考える，いわゆる「オブジェクト指向」と呼ばれるソフトウェアの概念がコンピュータの世界に持ち込まれ，注目されている。

【０００４】このようなオブジェクト指向によるソフトウェアの構築方法が，いろいろ研究・開発されているが，従来，例えば画像システム等といった特定応用分野のシステムを，オブジェクト指向を取り入れて容易に記述し，構築できるような言語処理システムの研究・開発は，まだ十分なされていない。

【０００５】

【発明が解決しようとする課題】画像システムやマルチメディアを扱うシステム等，特定応用分野のシステムを開発するとき，従来の汎用言語による開発では，ソフトウェアの生産性や信頼性の面から考えて十分とは言い難く，またソフトウェア・パッケージとしてもカスタマイズが困難であり，さらにまたユーザ・フレンドリなシステムの構築が難しいという問題がある。

【０００６】本発明は上記問題点の解決を図り，特定応用分野のシステムをオブジェクトネットワークとしてモデル化し，それを記述したものによって目的とする処理を行わせる言語処理システムを提供することにより，ソフトウェアの生産性，柔軟性を向上させ，かつユーザ・フレンドリなシステムの構築を容易にできるようにすることを目的としている。

【０００７】

【課題を解決するための手段】図１は，本発明の原理説明図である。図１において，１０は分野記述言語によるシステム記述，１１はトランスレータ，１２は実行システム，１３はプロセスの並行処理のための制御を行うプロセス構築管理機構，１４は名詞オブジェクトを管理する名詞オブジェクト管理機構，１５は動詞オブジェクトの実行制御機能を持つ動詞オブジェクト制御機構，１６はオブジェクトネットワークを定義するオブジェクト管理情報，２０はオブジェクトネットワーク，２１は名詞オブジェクト，２２は動詞オブジェクト，２３は制約条件，２４は総称的関数，２５は具体的関数を表す。

【０００８】本発明は，操作対象のデータおよびそれらのデータに対する操作手段をオブジェクトとして管理し処理する計算機システムにおいて，そのオブジェクトを名詞オブジェクト２１－１，２１－２，…と，動詞オブジェクト２２－１，２２－２，…とに大分類し，名詞オブジェクトをノード，動詞オブジェクトをブランチとして表現したオブジェクトネットワーク２０を，参照モデルとし，この参照モデルにおいて，あるノードに存在する名詞オブジェクトに動詞オブジェクトとしての関数の内容を作動させると，その動詞オブジェクト名に対応す

るブランチの方向にある目的対象となっている名詞オブジェクトが得られるようにしている。

【0009】名詞オブジェクトについては，普通名詞に対応する集合オブジェクト21aと，固有名詞に対応する個別オブジェクト21bとを有し，それらを名前づけまたは参照指示機能によって区別する名詞オブジェクト管理機構14を持つ。

【0010】名詞オブジェクト管理機構14は，名詞オブジェクトを修飾する形容詞としての制約条件を，名詞オブジェクトに付加し，オブジェクトとして管理する修飾管理機構を持ち，この修飾管理機構は，具体化された個別オブジェクトが属する名詞オブジェクトの持つ性質とそれを修飾する制約条件に対する充足性を判定する手段を持つ。

【0011】動詞オブジェクト22として，あるオブジェクトにデータなどの制約条件と併合した演算を施し，目的対象オブジェクトとして名詞オブジェクトを具体化する演算動詞と，集合オブジェクトを構成する個別オブジェクト群に作動して，新しい関係を持った名詞オブジェクトを生成する関係付加動詞と，目的名詞オブジェクトに，制約条件の性質を規定する前置詞と併合する名詞オブジェクトによって具体的制約条件を与え，制約を満足するように目的名詞オブジェクトに対し演算を行う制約演算動詞の少なくとも3種類の機能が存在し，これらが文単位で判定処理されるように構成されている。

【0012】動詞オブジェクトについて，総称的(generic) 関数24と，具体的(actual)関数25の種別を持つ。具体的関数25は，目的対象としての名詞オブジェクトに対し，実際に実行処理が可能な関数である。総称的関数24は，制約条件23が付けられることによって具体的関数25に変換される。動詞オブジェクト制御機構15は，総称的関数24から具体的関数25への変換制御を行う。

【0013】この動詞オブジェクト制御機構15は，総称的関数24に対応する動詞オブジェクトに対する制約条件23として，現在処理の実行が終了した名詞オブジェクト名を状態名として制約条件化する機能を持ち，これによって実行可能な具体的関数25を得るようになっている。

【0014】また，関数実行管理機構を有し，この関数実行管理機構は，動詞オブジェクトを関数として作動させるに当って，作動開始前，作動中，終了後の制約条件を付加し，各時点でこれらの妥当性を検定する手段を持つ。総称的関数24の制約条件は総称的条件で，これに加えて具体的制約条件が規定されて具体的関数25に変換される。このシステムの特徴として，以上の制約条件に関し，変換過程において逐次的に妥当性判定と変換のための変換手続の生成を行う制御機能を持つ。

【0015】オブジェクトネットワーク20中の普通名詞に対応する集合オブジェクトは，特定化された固有名詞に対応する個別オブジェクト集合を代表するので，集合オブジェクト名を持つ名詞オブジェクトに，データとしての制約を他から与えると，個々の個別オブジェクトが生成される。このために，参照モデルとしてのオブジェクトネットワーク20は，構造を示すネットワークであり，この中の集合オブジェクト状態から，特定の具体的関数の実行によって個別オブジェクトを生成し，それらの要素間関係を表現し，要素ネットワークを構成する。要素ネットワークでは，個別オブジェクトについての実行要求によって実際に処理を行う。プロセス構築管理機構13は，これらのネットワークの生成および管理を行う。

【0016】このプロセス構築管理機構13において，処理順序に関する制約条件については，結合されているオブジェクトネットワーク20中の動詞オブジェクトの方向性によって規定されると同時に，次にそのオブジェクトが実行可能な状態であることを示す可能性(possibility) トークンの存在する動詞オブジェクトが並行処理可能であることを判定し，この動詞オブジェクトに対する作動要求を受付けて具体的に実行を行わせるようなプロセスの並行処理のための制御が行われる。

【0017】構造を規定する部分を持つオブジェクトネットワーク20での動詞オブジェクトが作動して対応する要素ネットワークが作られ，そこに要素オブジェクトに関するデータが制約条件として付加される。このプロセスの実行は，動詞オブジェクトについてのシステム内外からの要求と，処理順序に関する制約のもとに，プロセス構築管理機構13の指示制御のもとに並列実行されるように構成されている。

【0018】プロセス構築管理機構13は，参照モデルとなる構造ネットワークに基づいて，新しく構造ネットワークを生成し，参照モデルとしてのオブジェクトネットワーク20の制約条件を，他システムの要求に従って変更改定し，新しく生成されたオブジェクトネットワークの制約条件を規定し，その結果完成した構造ネットワークが行うべき処理順序に基づいて要素ネットワークへの要求を他システムから受付け可能とすることにより，要素ネットワークへ制御権限を委託し，その結果処理が完了したことの受付けをし，構造ネットワークを消滅させる。この動作が同時に要素ネットワークの完成となるようにされ，処理結果が他システムへ表示される。

【0019】プロセス構築管理機構13から構造ネットワークの生成処理要求，その結果の応答，また構造ネットワークから要素ネットワークの生成処理要求および応答というように，要求(request) と応答(respond) とを対として，下層ネットワークへの要求，上層ネットワークへの応答というように，要求に対し，対応する関数の実行を行う関数実行管理機構がサービスを行う。この関数実行管理機構は，総称的関数から具体的関数への変換，実行上の妥当性検定の指示制御を行う。その結果不

備であれば新しい要求と応答との対を，逆に上層ネットワークへ向けて発行し，完備させる。完備していれば実行し，上層ネットワークへの応答の処理を行う。

【００２０】以上のオブジェクトネットワークによる言語処理システムのインプリメントは，例えば次のように実現される。図１の（イ）に示す分野記述言語によるシステム記述１０によって，（ロ）に示すオブジェクトネットワーク２０の定義，動詞オブジェクトに対応する総称的関数２４，具体的関数２５，制約条件２３の定義，また必要に応じてマンマシンインタフェースのためのウインドウ定義等を行う。この定義体をトランスレータ１１にかけて，内部データ表現で表されるオブジェクト管理情報１６に変換し，また定義された関数群を実行システム１２に組み込むことにより，自動的に特定応用分野のシステムが構築されるようになっている。

【００２１】

【作用】システムの設計者とユーザの両方の立場から，オブジェクト指向により，システムをデータ中心に構築するのが望ましい。システム内部での処理を，データの段階的な変換として捉え，各段階での変換を関数と考える。ここで，データの各段階をノードとし，関数をブランチとすると，これをネットワークとみなすことができる。

【００２２】すなわち，ソフトウェアシステムをネットワークで表現すると，ノードとしてのシステムの構成要素は，名詞オブジェクト（データ）であり，要素の接続関係であるブランチは，動詞オブジェクトである。動詞オブジェクトは，ノード間の変換関数として表現され，変換関数もネットワークで表現される。そして，ネットワーク自身もオブジェクトとみなすことができ，上位のネットワークの関係関数である。つまり，ネットワーク階層構造が存在する。

【００２３】以上のことから，本発明では，システムの構成要素をオブジェクトとして捉え，オブジェクトを名詞オブジェクトと動詞オブジェクトとに分類し，それらによって構成されるオブジェクトネットワーク２０を参照モデルとして，目的対象の処理を行わせる。

【００２４】

【実施例】図２は，本発明に係るオブジェクトネットワークの参照モデルの例を示す。ここでは，特定分野として，画像システムを例にして説明する。

【００２５】図２の（ロ）に示すように，ディスプレイ画面上で，画像を描画する場合を考える。最初，何もない状態(a) であり，マウス等によって点が指定されることにより，点が描画された状態(b) になる。さらに，いくつかの点が指定されて，点列が形成された状態(c) に遷移する。

【００２６】このような点や点列などのデータを，名詞オブジェクトとして捉えると，ある名詞オブジェクト（例えばpoint)に，関数としての動詞オブジェクト（例えばlist points)が作動することにより，目的とする名詞オブジェクト（例えばpointsequence）が得られると考えることができる。すなわち，図２の（イ）に示すように，描画対象のデータの各状態を，名詞オブジェクトとしてノードに置き，ノード間を結ぶブランチを動詞オブジェクトとして，オブジェクトネットワークを表現すれば，そのオブジェクトネットワークによって，そのまま描画のための画像システムの構造が形成できることになる。

【００２７】例えば，名詞オブジェクト(point）について，特定の座標値がまだ定まっていない抽象的な点の概念に相当するものは，普通名詞に対応する集合オブジェクトとして扱われ，実際に点(point）の座標値が付与されて個性を持つものは，その集合オブジェクトから生成された固有名詞に対応する個別オブジェクトとして扱われる。したがって，１つの集合オブジェクトから多数の個別オブジェクトが派生することになり，この意味で図１に示すような分野記述言語によるシステム記述１０によって定義されたオブジェクトネットワーク２０は，参照モデルである。

【００２８】図３は，本発明に係る名詞オブジェクト管理機構の構成例を示す。図１に示す名詞オブジェクト管理機構１４は，図３に示すように，個別オブジェクト２１ｂに対して，ユーザまたはシステムが任意に名前を付けることができる名前づけ機能３２を有し，これにより付けた名前を管理する名前管理機能３３を有する。この名前によって，特定の個別オブジェクト２１ｂを他のものと区別することできるようになっている。また，特定の個別オブジェクト２１ｂについての参照を指示する参照指示機能３４によっても，他のものと区別することが可能である。

【００２９】また，名詞オブジェクト管理機構１４は，名詞オブジェクトを修飾する形容詞としての制約条件２３－１，２３－２を，名詞オブジェクトに付加し，オブジェクトとして管理する修飾管理機構３０を持つ。この修飾管理機構３０は，具体化された個別オブジェクト２１ｂが属する名詞オブジェクトの性質と，それを修飾する制約条件に対する充足性を判定する制約条件妥当性検査／制約条件付加機能３１を持つ。

【００３０】制約条件妥当性検査／制約条件付加機能３１によって，普通名詞に対応する集合オブジェクト２１ａに対する制約条件は，それに属する固有名詞としての個別オブジェクト２１ｂの制約条件ともなり，さらに妥当性のある制約条件を付加しうる。

【００３１】図４は，本発明に係る動詞オブジェクトの動詞の種類を示している。(a) に示す演算動詞は，オブジェクトＡをオブジェクトＢに変換する関数の要素である。図２の例において，名詞オブジェクト(None)から座標値のデータを制約として名詞オブジェクト(point）を生成する要素がこれに当たる。

【００３２】(b) に示す関係付加動詞は，固有名詞オブジェクト群の集合オブジェクトＳに，ある関係を与えて新しいオブジェクト（関係オブジェクトという）を生成する関数の要素である。図２の例において，point の順序集合である名詞オブジェクト(point sequence)に，point の接続条件を与えて，新しい関係を持った名詞オブジェクト(line segment)を生成する要素がこれに当たる。

【００３３】(c) に示す制約演算動詞は，オブジェクトＡからオブジェクトＣへの変換を行うとき，制約条件としてオブジェクトＢが与えられるような関数の要素である。図２の例では，点に付与される輝度属性を示す名詞オブジェクト(luminance onthe point)を，点の順序集合に対する輝度属性を示す名詞オブジェクト(luminance on the point sequence) に変換するとき，点の順序集合を定める名詞オブジェクト(point sequence)が制約条件となるが，このような場合が制約演算動詞に相当する。

【００３４】図５は，本発明に係る総称的関数と具体的関数の関係説明図である。動詞オブジェクトの具体化には，総称的関数２４と制約条件２３とから，具体的関数２５を生成する翻訳生成機能５０を設けることにより，ユーザに対し，総称的関数２４だけを意識させればよい構造とする。ユーザは，実際の処理手続きに対応する具体的関数２５を覚える必要がなくなり，ユーザ・フレンドリ性の増大を図ることができる。

【００３５】例えば，send(X) と名付けられた総称的関数は，send(X) to D で，DにXを送る。この送る手段として制約条件を付けることができる。また，総称的動詞オブジェクトに対する制約条件として，現在実行中の名詞オブジェクト名がシステムの現状態を表すことを積極的に利用する。例えば，図５の（ロ）に示すように，現在のシステムの実行状態である名詞オブジェクト名がNoneであるとき，それを制約条件とすることにより，総称的関数(draw)２４を，具体的関数(set point) ２５に変換する。

【００３６】例えば，図２の（イ）に示すオブジェクトネットワークにおいて，set point,list points，generate curve は，すべて具体的関数であるが，これらを１つの総称的関数(draw)で表すことができる。ユーザは，set point，list points，generate curveといった具体的関数を覚える必要はなく，すべてdrawとして取り扱うことができる。drawが，set point，list points，generate curveのどれを意味するかは，現在のシステムの実行状態が制約条件として付加されることにより，自動的に決定される。

【００３７】すなわち，総称的関数(draw)＋制約条件(None)→具体的関数(set point) であり，総称的関数(draw)＋制約条件(point) →具体的関数(list points) である。図６は，本発明の実施例に係る動詞オブジェクトの実行管理説明図である。

【００３８】動詞オブジェクトの具体的実行に際しては，さらに作動の開始前制約条件２３ａ，作動中制約条件２３ｂ，終了制約条件２３ｃが付加され，これらが考慮される。関数実行管理機構６０は，総称的関数に対応する動詞名の指定による作動要求があると，開始前制約条件２３ａを付加し，他の制約条件と合わせて，妥当性の検査および具体的関数への変換を行う。そして，具体的関数の実行６１に移るが，実行中においても，逐次的に作動中制約条件２３ｂを付加し，システム動作の妥当性検定機能や推論機能に役立てる。さらに，実行終了時には，終了制約条件２３ｃを付加し，妥当性検定機能に役立てる。

【００３９】例えば，円弧を描く場合，少なくとも３点の座標値が定められていなければならない。もし，２点の座標値しか決まっていなかったとすると，円弧を描く関数の実行は矛盾を生じさせることになる。しかし，本実施例においては，関数実行管理機構６０における妥当性検定機能によって事前に検査し，矛盾の発生を回避することができる。この機構により，必要に応じて３点目の座標値の入力をユーザに要求する関数を自動的に起動させることもできる。

【００４０】図７は，本発明に係る並行プロセス構築の層構造の例を示す。ユーザから提起される多様な要求に対し，一般にシステムは機能を適当なレベルで基本的なモジュールに分け，それらを必要な順序で縦続的に，あるいは並行的に組み合わせ，積み上げることにより処理を進める形態が採られる。各要求に対する処理のしかたと，ユーザ要求の相続く発生とから，システムは処理の階層構成と各層における種々の並行処理を必要とする。

【００４１】本発明によって構築されるシステムでは，このような処理は，名詞オブジェクトと動詞オブジェクトとからなるオブジェクトネットワーク上の移動として考えられる。このオブジェクトネットワークにおいて，複数のプロセスが並行処理される場合には，オブジェクトネットワーク上の移動に関する制御が必要となる。以下，このオブジェクトネットワークと並行プロセスの構築について説明する。

【００４２】独立な制御を受ける単位をプロセスとする。複数個のプロセスの進行が考えられるシステムが，並行プロセスシステムである。特に，複数個のプロセスが１つのまとまった仕事を遂行する場合，ある操作の実行や同一資源へのアクセスに際し，プロセス間に相互作用が生じ，順序等を規定する必要がでてくる。並行プロセスをオブジェクトネットワーク上で実現させるには，ノード間の移動を制御することが必要になる。そのため，図７に示すようなシステムの階層化を図り，プロセス構築管理機構１３によって，プロセスの制御を行う。

【００４３】階層として，基本的に次の４階層がある。

(1) プロセス構築層：集合オブジェクトの生成要求を受

付け，個別オブジェクトを生成するプロセスを決定する。

【００４４】(2) 構造ネットワーク層：参照モデルとしてのオブジェクトネットワークを示し，処理順序を制御する。

(3) 要素ネットワーク層：個別オブジェクトを生成する。

【００４５】(4) 関数実行層：実際に関数を実行する。これらの層の間では，要求と応答の対によって制御の受け渡しが行われる。プロセス構築管理機構１３は，外部ユーザからのシステムへの要求を受け，そのオブジェクトの処理順序を決定し，次段の層の構造ネットワーク７１－１，７１－２を生成する。構造ネットワーク７１－１では，プロセス構築管理機構１３によって決定された処理順序に従って，要素ネットワーク７２を生成する。ネットワーク自身もオブジェクトとみなし，さらに詳細な処理順序を制御するためのサブプロセス構築管理機構７３への要求を行うこともできる。

【００４６】要求を受けたサブプロセス構築管理機構７３では，同様にサブ構造ネットワーク７４への生成処理要求，サブ構造ネットワーク７４では，サブ要求ネットワーク７５への生成処理要求を行い，順次，下位層からの応答を受ける。

【００４７】並行プロセスを実現するための実行順序の具体化として，ペトリネットワークにおけるトークンの制御方式を用いることができる。図８は，その制御方式を示している。

【００４８】図８において，例えばオブジェクトＡをpoint に対するluminance data値（輝度データ値）とすると，点としてのpoint をidentifyすることが，可能性トークンを発生させ，これとオブジェクトＡからの実行トークンとがそろうことにより，実行要求が発せられる。

【００４９】図９は，要素オブジェクトを生成する例を示している。構造を規定する部分を持つ構造ネットワーク７１での動詞オブジェクトが作動して，要素ネットワーク７２が作られる。そこに，要素オブジェクトに関するデータが制約条件として付加される。この参照生成の際における処理順序の制御に関する制約条件は，他システムからの要求に従って変更・改定することができる。要素オブジェクトＡ１，Ａ２は処理順序の制約が異なる。

【００５０】次に，特定応用分野への適用例として，カラー画像を描画するための画像システムに本発明を適用した例を説明する。画像システムに限らず，イベント駆動やデータ駆動によって，システム内の状態を変化させるような動作を行う各種のシステムについて，同様に本発明を適用することができる。

【００５１】まず，図１に示す分野記述言語によるシステム記述１０に相当する画像システム記述言語の構文と意味について説明する。本実施例による画像システム記述言語は，オブジェクトネットワークを記述するネットワーク記述部，関数を記述する関数記述部，オペレータとの入出力インタフェースのためのウインドウを定義するウインドウ記述部を持つ。構文は，文脈自由文法であるので，ＢＮＦ（Backus-Naur Form）により記述可能である。構文に関係する基本語句は以下のとおりである。

【００５２】(1) 文字

この画像システム記述言語で使用される文字は，英大文字，英小文字，数字の３種類である。英大文字は，定義文の記述や識別子に用いられ，英小文字は，識別子に用いられる。数字は，数値の表現や識別子の中で使われる。英大文字，英小文字，数字をＢＮＦ記法で表す。

【００５３】<英小文字>：：＝ａ｜ｂ｜ｃ｜ｄ｜ｅ｜ｆ｜……｜ｘ｜ｙ｜ｚ｜＿

<英大文字>：：＝Ａ｜Ｂ｜Ｃ｜Ｄ｜Ｅ｜Ｆ｜……｜Ｘ｜Ｙ｜Ｚ｜＿

<数字>　：：＝１｜２｜３｜４｜５｜６｜７｜８｜９｜０

(2) 数値

数値は，座標や大きさなどを表すために使われる。ＢＮＦ記法で表すと，

<数値>：：＝<数値><数字>｜<数字>　となる。

【００５４】(3) 識別子

ネットワーク名やウインドウ名などの名前の表現のために使われる。識別子の規則として先頭の文字は，英文字でなくてはならない。識別子は，英小文字と数字だけ，英大文字と数字だけ，英大文字と英小文字と数字混在の３種類がある。

【００５５】

<英小文字&数字>：：＝<英小文字>｜<数字>

<英大文字&数字>：：＝<英大文字>｜<数字>

<英文字&数字>　：：＝<英小文字>｜<英大文字>｜<数字>

<小文字識別子>：：＝<小文字識別子><英小文字&数字>｜<英小文字>

<大文字識別子>：：＝<大文字識別子><英大文字&数字>｜<英大文字>

<識別子>：：＝<識別子><英文字&数字>｜<英小文字>｜<英大文字>

(4) ネットワーク記述部

ネットワーク記述部は，ネットワーク定義文(DEF-NET文）により，オブジェクトネットワークを記述する。ネットワーク定義文の構文は，次のようになる。

【００５６】

DEF-NET [<ネット名>] {<ネットワーク定義>}

<ネットワーク定義>::＝ENTITY [<項目ネット名>] {<項目リスト>;} <属性定義>

<属性定義>::＝<属性定義> ATTRIBUTE [<属性ネット名>] {<属性リスト>;} ｜ ATTRIBUTE [<属性ネ

ット名>]〔<属性リスト>;〕
<項目リスト>::=<項目オブジェクト>;<項目リスト>|<項目オブジェクト>;
<属性リスト>::=<属性オブジェクト>;<属性リスト>|<属性オブジェクト>;
<項目オブジェクト>::=<データクラス名>:<総称的関数リスト>
<属性オブジェクト>::=<属性名>:<総称的関数リスト>
<総称的関数リスト>::=<総称的関数リスト>,
,<総称的関数名> －> <方向>
|<総称的関数名> －> <方向>
<方向>::=ＵＰ|ＤＯＷＮ|ＥＮＤ|<データクラス名>!<項目ネット名>@<ネット名>|<属性名>!<属性ネット名>@<ネット名>
<ネット名>::=<識別子>
<項目ネット名>::=<識別子>
<属性ネット名>::=<識別子>
<データクラス名>::=<大文字識別子>
<属性ネット名>::=<大文字識別子>
<総称的関数名>::=<大文字識別子>
このネットワーク定義文の意味を，箇条書きに自然言語で表現すると，次のとおりである。
【００５７】(a) この定義文は，オブジェクトネットワークを定義する。
(b) オブジェクトネットワークは，１つの項目(entity)ネットワークと複数の属性(attribute) ネットワークから構成される。
【００５８】(c) 項目ネットワークは，データクラスとそれに対応する総称的関数リストで構成される。
(d) 属性ネットワークは，データクラスの属性名とそれに対応する総称的関数リストで構成される。
【００５９】(e) 総称的関数リストは，総称的関数とその関数の実行後の行き先を“－>”で繋いだリストで構成される。
(f) 行き先は，１つ上位のデータクラスへの移動を意味する“ＵＰ”か，１つ下位のデータクラスへの移動を意味する“ＤＯＷＮ”か，終了を意味する“ＥＮＤ”か，ネットワーク内の絶対的な位置を指定する。指定の書式は，データ名（データの属性名）!項目ネットワーク名（属性ネットワーク名）@ネットワーク名である。
【００６０】(5) 関数記述部
関数記述部は，総称的関数定義文(DEF-GENERIC-FUNC文）により，総称的関数を記述し，具体的関数定義文(DEF-ACTUAL-FUNC文）により，具体的関数を記述する。
【００６１】(5-1) 総称的関数定義文の構文を，ＢＮＦ記法で示す。
DEF-GENERIC-FUNC [<総称的関数名>]〔<具体的関数リスト>;〕
<具体的関数リスト>::=<具体的関数リスト>;<制約付き具体的関数>|<制約付き具体的関数>
<制約付き具体的関数>::=<制約条件>:<具体的関数名>
<制約条件>::=<制約条件><論理演算子><表現>|<表現>
<表現>::=<システム変数><関係演算子><値>
<論理演算子>::=＆＆|‖
<関係演算子>::= ==|!=|<|>|<=|>=
<システム変数>::= CURRENT-NET|CURRENT-PART|CURRENT-STEP|DIRECTION
<総称的関数名>::=<大文字識別子>
この総称的関数定義文の意味を箇条書きで表すと，次のとおりである。
【００６２】(a) 総称的関数定義文は，総称的関数を定義する。
(b) 総称的関数は，制約条件が真であった場合，対応する具体的関数が選択されるよう定義される。
【００６３】(c) 制約条件は，システム変数とその値を関係演算子で接続した式を，論理演算子で接続して表す。
(d) システム変数は，CURRENT-NET が現在のネットワークを表し，CURRENT-PARTが選択されている項目・属性ネットワークを表し，CURRENT-STEPがネットワーク上の位置を表し，DIRECTION がネットワーク上を進む方向を表す。
【００６４】(5-2) 具体的関数定義文の構文を，ＢＮＦ記法で表す。
DEF-ACTUAL-FUNC [<具体的関数名>] :<出力クラス名>,<入力クラス名リスト>]
{ ／＊ Ｃ言語プログラム ＊／ }
<出力クラス名>::=<データクラス名>
<入力クラス名リスト>::=<入力クラス名リスト>／<入力クラス>|<入力クラス>
<入力クラス>::=<データクラス名>:<要求形式>
<要求形式>::= ALL|CURRENT |<具体的関数名>
<具体的関数名>::=<小文字識別子>
<データクラス名>::=<大文字識別子>
この具体的関数定義文の意味を箇条書きで表すと，次のとおりである。
【００６５】(a) 具体的関数定義文は，具体的関数を定義する。
(b) 具体的関数は，関数の結果としての出力クラスと，関数の引数としての入力クラスリストを定義する。
【００６６】(c) 入力クラスリストは，入力クラスとそのクラスの要求形式を指定する。要求形式は，そのクラスのすべてのデータを要求する“ＡＬＬ”と，１つ前に実行されたそのクラスのデータを要求する“ＣＵＲＲＥＮＴ”と，そのクラスのデータを生成する具体的関数の実行を要求する<具体的関数名>がある。
【００６７】(6) ウインドウ記述部

ウインドウ記述部は，ウインドウクラス定義文(DEF-WIN-CLASS文）により，データウインドウクラスを記述し，データウインドウ定義文（DEF-DATA-WIN文）により，データウインドウを記述し，選択ウインドウ定義文（DEF-SELECT-WIN文）により，選択ウインドウを記述する。

【００６８】データウインドウは，データウインドウクラスのオブジェクトとして生成され，指定された名詞オブジェクトの表示および消去ができ，マウス等により指定された座標データの入力ができる機能を持つウインドウである。選択ウインドウは，システムが処理を行うとき，ユーザの判断が必要になる場合があるが，そのようなとき，ユーザに対し判断を要求するためのウインドウである。

【００６９】(6-1) ウインドウクラス定義文の構文をＢＮＦ記法で表す。

DEF-WIN-CLASS [<ウインドウクラス名>：<バースイッチ>,<表示域の幅>，<表示域の高さ>]
<ウインドウクラス名>::=<識別子>
<バースイッチ>::= ＯＮ｜ＯＦＦ
<表示域の幅>::=<数値>
<表示域の高さ>::=<数値>

この意味は，次のとおりである。

【００７０】(a) データウインドウクラスを定義する。

(b) データウインドウクラスは，生成されるデータウインドウが，スクロールバーを持つかどうかを指定するスイッチと，仮想的画面である表示域(canvas)の縦横の大きさを指定する。

【００７１】(6-2) データウインドウ定義文の構文をＢＮＦ記法で表す。

DEF-DATA-WIN [<データウインドウ名>：<ウインドウクラス名>,<初期値ｘ>，<初期値ｙ>，<初期幅>，<初期高>]
<データウインドウ名>::=<識別子>
<初期値ｘ>::=<数値>
<初期値ｙ>::=<数値>
<初期幅>::=<数値>
<初期高>::=<数値>

この意味は，次のとおりである。

【００７２】(a) ウインドウクラスのインスタンスとしてデータウインドウを定義する。

(b) データウインドウの初期表示位置，縦横の大きさを＊定義する。

(6-3) 選択ウインドウ定義文の構文をＢＮＦ記法で表す。

【００７３】DEF-SELECT-WIN [<選択ウインドウ名>]{<選択ウインドウリスト>}
<選択ウインドウ名>::=<識別子>
<選択ウインドウリスト>::=<選択ウインドウリスト>；<選択ウインドウ定義>｜<選択ウインドウ定義>
<選択ウインドウ定義>::= menu(<表示項目リスト>)｜key(<メッセージ>)
<表示項目リスト>::=<表示項目リスト>／<表示項目>｜<表示項目>
<表示項目>::=<英文字&数字>
<メッセージ>::=<英文字&数字>

この意味は，次のとおりである。

【００７４】(a) 選択ウインドウを定義する。

(b) 選択ウインドウは，メニュータイプかキータイプかのどちらかを指定できる。

【００７５】(c) メニュータイプの場合には，表示する項目（item）を記述する。

(d) キータイプの場合には，メッセージを記述する。

図１０は，本発明の実施例による画像描画用オブジェクトネットワークの例を示す。

【００７６】例えば，図１０に示すような画像描画用オブジェクトネットワークに従って動作する画像システムを構築するとする。このオブジェクトネットワークでは，画像データが，ポイント，ライン，リージョンなどの数段階のノードに分けられ，あるデータクラスのデータを別のデータクラスのデータに変換させる動詞オブジェクトとして，ポイント生成，ライン生成，リージョン生成などのいくつかの関数が存在する。これらの関数は，DRAW，SEGMENTALIZE等といった総称的関数によって表現される。

【００７７】画像描画において重要である色情報は，各画像データの属性として定義され，この属性に対してもオブジェクトネットワークが定義される。図１０に示すオブジェクトネットワークを，上述したネットワーク定義文の記述文法によって定義すると，以下のようになる。

【００７８】

```
DEF-NET [Painting] {
    ENTITY [Outline ] {
      NONE        : DRAW -> UP ;
      POINT       : DRAW -> UP ;
      POINT-SEQ   : DRAW -> UP ;
      LINE-SEG    : DRAW -> UP ;
      LINE        : SEGMENTALIZE -> UP , DRAW -> POINT ;
      REGION-SEG  : SEGMENTALIZE -> UP ;
      REGION      : DRAW -> POINT, PUT -> POINT-COLOR ! Color ;
```

```
  }
  ATTRIBUTE [Color ] {
    NONE            : PUT－＞ UP ;
    POINT-COLOR     : PUT－＞ UP ;
    POINT-SEQ-COLOR : PAINT－＞ UP ;
    LINE-SEG-COLOR  : PAINT－＞ UP ;
    LINE-COLOR      : PAINT－＞ UP ;
    REGION-SEG-COLOR: PAINT－＞ UP ;
    REGION-COLOR    : PUT－＞ POINT-COLOR;
  }
}
```

この描画用オブジェクトネットワークのネットワーク名は“Painting”であり，項目ネットワークとして“Outline”，属性ネットワークとして“Color”が定義されている。このオブジェクトネットワークでは，LINE，REGIONにおいて複数の関数が定義されている。LINEの場合，DRAW－＞ POINTが定義されることにより，総称的関数DRAWが実行された後，POINT に移動するようにされている。

【００７９】これにより，ループ構造を記述することができ，複数のLINEを描画できる。このループから抜け出るためには，LINEにおいてSEGMENTALIZEを選択する。REGIONでは，DRAW－＞ POINTと，PUT －＞ POINT-COLOR！Color が定義されている。DRAW－＞ POINTは，DRAW実行後，POINT へ移動することを表し，複数の輪郭を描画できる。PUT －＞ POINT-COLOR！Color は，PUT 実行後，"Color" ネットワークのPOINT-COLOR へ移動することを示している。これにより，輪郭生成終了後，直接，色情報生成ネットワークへ移動することができる。

【００８０】動詞オブジェクトの内容となる描画関数は，総称的関数と具体的関数が，例えば次のように定義される。

(1) 総称的関数の定義

```
DEF-GENERIC-FUNC [ DRAW ] {
CURRENT-NET == "Painting"&& CURRENT-PART == "Outli
ne"&&
CURRENT-STEP == "NONE"&& DIRECTION == "UP" : set-p
p;
CURRENT-NET == "Painting"&& CURRENT-PART == "Outli
ne"&&
CURRENT-STEP == "POINT"&& DIRECTION == "UP" : creat
e-ps ;
CURRENT-NET == "Painting"&& CURRENT-PART == "Outli
ne"&&
CURRENT-STEP == "POINT-SEQ"&& DIRECTION == "UP" : c
reate-ls ;
CURRENT-NET == "Painting"&& CURRENT-PART == "Outli
ne"&&
CURRENT-STEP == "LINE-SEG"&& DIRECTION == "UP" : c
reate-l ;
CURRENT-NET == "Painting"&& CURRENT-PART == "Outli
ne"&&
CURRENT-STEP == "LINE"&& DIRECTION == "POINT" : set
-pp ;
CURRENT-NET == "Painting"&& CURRENT-PART == "Outli
ne"&&
CURRENT-STEP == "REGION"&& DIRECTION == "POINT" : s
et-pp ;
```

総称的関数は，上述したシステム記述の文法に従って，具体的関数への変換条件を記述することで定義される。ここでは，総称的関数DRAWを例に，その定義例を示したが，SEGMENTALIZE,CONNECT,PUT,PAINT等についても，同様に定義される。

【００８１】DRAWは，輪郭生成を行う総称的関数であり，具体的関数であるset-pp，create-ps，create-ls，create-lのいずれかに対応する制約条件が与えられることにより，具体的関数に変換される。例えば，制約条件として，現在のネットワークCURRENT-NET が "Painting” であり，選択されている項目ネットワークCURRENT-PARTが "Outline"であり，ネットワーク上の位置CURRENT-STEPが"NONE"であり，ネットワーク上の進む方向DIRECTION が "UP" であれば，総称的関数DRAWは，点の座標データを設定する具体的関数set-ppに変換される。

【００８２】(2) 具体的関数の定義

この画像システムにおける具体的関数は，次のように定義される。

```
DEF-ACTUAL-FUNC [set-pp : POINT, COORDINATES :draw
1]
DEF-ACTUAL-FUNC [create-ps : POINT-SEQ, POINT : cu
rrent]
DEF-ACTUAL-FUNC [create-ls : LINE-SEG, POINT-SEQ :
 current ]
DEF-ACTUAL-FUNC [create-l : LINE, LINE-SEG : curre
nt ]
DEF-ACTUAL-FUNC [create-rs : REGION-SEG, LINE : cu
rrent]
DEF-ACTUAL-FUNC [create-r : REGION, REGION-SEG : c
urrent ]
DEF-ACTUAL-FUNC [create-de : REGION, DESSIN-ELEMEN
```

T : current]
DEF-ACTUAL-FUNC [put-color-pp : POINT,
COORDINATES :draw1/l-diagram/c-diagram]
DEF-ACTUAL-FUNC [put-color-ps : POINT-SEQ, POINT : current]
DEF-ACTUAL-FUNC [paint-color-ls : LINE-SEG, POINT-SEQ : current]
DEF-ACTUAL-FUNC [paint-color-l : LINE, LINE-SEG : current]
DEF-ACTUAL-FUNC [paint-color-rs : REGION-SEG, LINE : current]
DEF-ACTUAL-FUNC [paint-color-r : REGION, REGION-SEG : current]
ここで，画像データクラスCOORDINATES について補足説明する。COORDINATESは，座標を表すクラスで，画像データクラスの中で最も上位のクラスである。よってこのクラスは，具体的関数の出力クラスに書かれることはない。入力クラスに書かれる場合，他の画像データクラスは，＜要求形式＞が書かれるが，COORDINATES のみ，座標データ入力を要求するデータウインドウ名を指定する。座標データ要求ウインドウ名は，複数指定可能である。

【００８３】具体的関数put-color-ppの入力データクラス指定では，COORDINATES の要求ウインドウ名として，データウインドウdraw1，l-diagram，c-diagram が指定されている。これにより，この３つのデータウインドウから並行に座標データの入力が可能になる。

【００８４】本実施例では，具体的関数の具体的な処理の記述は，Ｃ言語によって行っている。具体的処理内容は，以下のとおりである。

set-pp　　：主要点のセット。

【００８５】create-ps　：ポイントシーケンスを生成。
create-ls　：ラインセグメントを生成。
create-l　：ラインを生成。

【００８６】create-rs　：リージョンセグメントを生成。
create-r　：リージョンを生成。
create-de　：デッサンエレメントを生成。

【００８７】put-color-pp　：色度情報の主要点への設定。
put-color-ps　：色度情報のポイントシーケンスへの設定。
paint-color-ls：ラインセグメント上での色度のペイント。

【００８８】paint-color-l　：ライン上での色度のペイント。
paint-color-rs：リージョンセグメント上での色度のペイント。
paint-color-r　：リージョン上での色度のペイント。

【００８９】この他に，ウインドウ関係の定義も行われ，画像システムが記述される。このような定義体によるシステム記述は，図１に示すようなトランスレータ１１によって実行形式データに変換される。図１１は，そのトランスレータ１１のフローチャートを示している。以下の処理を行う。

【００９０】① システム記述（ソースプログラム）を入力する。

② システム記述言語の文法に従って，ソースプログラムの構文解析を行い，必要な情報をデータベースに整理する。

【００９１】③ データベースから実行システム（ランタイムシステム）用のデータを生成する。

実行システムの構造は，本実施例では，図１２に示すようになっている。図１２において，１２０は画像システムカーネル，１２１はオブジェクト間インタフェース（ＩＦＯ：Interface Function between Object），１２２はデータマネージャ，１２３は関数マネージャ，１２４はウインドウマネージャを表す。

【００９２】画像システムカーネル１２０は，実行システムの中心に存在し，トランスレータからのデータを基に，以下の処理を行う。

(1) オブジェクトネットワーク上でのデータ状態把握
描画プロセスの把握を行い，次に実行すべき関数を選択する。

【００９３】(2) イベントの解析
各ウインドウから送られてくるイベントを判別し，その結果に対応するシステム内のプロシジャをコールする。

【００９４】(3) 関数実行要求
総称的関数に制約条件を付加し，オブジェクト間インタフェース１２１に関数の実行を要求する。

【００９５】オブジェクト間インタフェース１２１は，画像システムカーネル１２０と，データ，関数，ウインドウを管理するマネージャ間を，要求（リクエスト）－応答（レスポンド）方式の標準的プロトコルにより，インタフェースをとる。標準プロトコルを用意することで，画像システムカーネル１２０には変更を加えずに，言語のカスタマイズが可能になる。

【００９６】オブジェクト間インタフェース１２１のプロトコルは，データマネージャ１２２，関数マネージャ１２３，ウインドウマネージャ１２４の各々に対しての３つに分けられている。

【００９７】データマネージャ１２２に対しては，データの読み書き，変更，削除などのデータ操作に関するプロトコルが用意されている。関数マネージャ１２３に対しては，総称的関数から具体的関数への変換，具体的関数の実行に必要とするデータの準備，具体的関数の実行要求プロトコルがある。ウインドウマネージャ１２４に対しては，データウインドウ，選択ウインドウ，メッセージウインドウ，オペレーションウインドウに対するプロトコルが用意されている。

【００９８】これらのプロトコルの実現のために，オブジェクト間インタフェース１２１は，カーネルインタフェース，関数サーバ，データサーバ，ウインドウサーバを持つ。

【００９９】カーネルインタフェースは，画像システムカーネルと内部の各サーバとのインタフェースである。このカーネルインタフェースは，ただ単にインタフェースの役目を行っているだけでなく，カーネルからの要求を解析し，解析した結果により，その要求を満足できるサーバに対し，処理要求を行い，処理を進める。

【０１００】関数サーバは，関数の起動要求に従い，要求のあった総称的関数名により，そのときシステムの状態を付加し，そのときに起動すべき具体的関数を実行させる。データサーバは，名詞オブジェクトについて，各クラスごとに登録，読み出し，修正，削除を行う。また名詞オブジェクトを代名詞によって参照する機能を持つ。ウインドウサーバは，ウインドウマネージャ１２４を介して，登録されているウインドウへの描画，入力などの入出力処理を行う。

【０１０１】実際のデータ，関数，ウインドウは，それぞれ各マネージャにより一括管理される。データマネージャ１２２は，画像データの管理を行う。関数マネージャ１２３は，関数の実行管理を行う。ウインドウマネージャ１２４は，ウインドウの管理およびウインドウへのオブジェクトの描画を行う。これらのマネージャは，一般に構築するシステムによりカスタマイズされるが，そのプロトコルは変更されないので，他に影響を与えずにカスタマイズが可能である。

【０１０２】図１３は，この実行システム（ランタイムシステム）のフローチャートを示している。

(a) イベントの駆動制御により，入力イベント等が発生すると，画像システムカーネル（以下，単にカーネルという）が起動される。

【０１０３】(b) カーネルは，イベントを解析する。

(c) イベントを判別した結果，関数リクエスト待ち状態のときのオブジェクトウインドウからのイベントであれば，処理(d) へ進む。座標データ入力待ち状態のときのデータウインドウからのイベントであれば，処理(j) へ進む。

【０１０４】(d) リクエストされた関数（動詞オブジェクト）を判別する。

(e) カーネルは，オブジェクト間インタフェース（ＩＦＯ）に要求し，総称的関数から具体的関数への変換を行う。

【０１０５】(f) さらにＩＦＯに，その関数が座標データを必要とするかどうか，必要とする場合，その座標データが保存されているかどうかを問い合わせる。

(g) 問い合わせた結果，座標データが必要ない場合には，処理(h) へ進み，座標データが必要である場合には，処理(i) へ進む。

【０１０６】(h) ＩＦＯに関数の実行を要求し，関数を実行する。その後，処理(a) により次のイベントを待つ。

(i) 座標データが必要な場合，カーネルを座標データ入力状態にし，次のイベントを待つ。

【０１０７】(j) イベントが，データの入力イベントであれば，そのデータが座標データか制御用データかを判別する。制御用データとは，例えばデータ入力終了を示すようなデータである。

【０１０８】(k) 制御用データの場合，処理(n) へ進み，座標データの場合，処理(l) へ進む。

(l) ＩＦＯに依頼し，座標データを保存する。

【０１０９】(m) 座標データが用意されることにより，関数の実行が可能であれば，関数を実行し，次のイベントを待つ。

(n) データ入力終了を示すような制御用データが入力されたならば，カーネルを関数リクエスト待ち状態にし，リクエスト（イベント）を待つ。

【０１１０】この画像システムでは，ユーザとのインタフェースとして，マルチウインドウを使用している。ウインドウは，その機能別に分割され，オペレーションウインドウ，データウインドウ，メッセージウインドウ，選択ウインドウの４種類がある。データウインドウ，選択ウインドウは，前述したシステム記述の定義文により，ユーザがカスタマイズできる。

【０１１１】図１４に，本発明の実施例におけるオペレーションウインドウの例を示す。図１４において，(a) はネットワークウインドウの制御，データウインドウの呼び出しおよびシステム全体の制御を行うためのコントロールウインドウ，(b)はオブジェクトネットワークを表示するネットワークウインドウを表す。各部の詳細は以下のとおりである。

【０１１２】(a) コントロールウインドウ

ａ１は，データウインドウ切り替えボタンである。このボタンをヒットとすると，登録されているデータウインドウ名のメニューが表示され，画面の最前列に呼び出したいデータウインドウ名を選択することができる。

【０１１３】ａ２は，ネットワーク切り替えボタンである。この画像システムでは，複数のオブジェクトネットワークを使用できるように設計されている。しかし，ネットワークウインドウは１つであるため，表示するネットワークを切り替える方式を採る。このボタンを押すと，登録されているネットワーク名のメニューが表示され，使用するネットワークを選択することができる。

【０１１４】ａ３は，システムボタンである。このボタンにより，システムの終了，状態の保存など，システム全体の制御を行うようになっている。ａ４は，ネットワーク名表示エリアである。現在選択されているネットワーク名を表示する。

【０１１５】ａ５は，項目・属性名表示エリアである。

現在選択されているネットワークの項目・属性ネットワーク名を表示する。

(b) ネットワークウインドウ

ｂ１は，項目・属性切り替えウインドウである。現在選択されているネットワークに含まれる項目・属性ネットワークを切り替える。一番左側に表示されているボタンが，項目ネットワーク名であり，１つだけ存在する。それより，右側に表示されているボタンは，属性ネットワーク名を示し，複数存在することがある。表示されている任意のボタンをヒットすると，選択されたネットワークがｂ２の表示ウインドウに表示される。

【０１１６】表示ウインドウｂ２は，オブジェクトネットワークを表示し，ネットワークをどこまで進んだかと，そこからどのようなオペレーションが行なえるかというシステムの状態を表現する。ネットワーク上の現在の位置を表す名詞オブジェクト名（例えばPOINT 等）は，赤文字で表示され，次に実行できる動詞オブジェクトの関数名（例えば上から２つ目のDRAW等）は，青枠で囲まれて，他と区別される。

【０１１７】ユーザは，マウスにより，青枠で囲まれた関数名をヒットするか，赤文字で表示された名詞オブジェクト名（データ名）の１つ下をヒットすることで，次に実行するオペレーションを要求することができる。ヒットされた関数は実行中であることをユーザに示すため，反転表示される。実行が終了すると，今まで赤文字で表示されていたデータ名と青枠で囲まれ反転表示されていた関数名が黒文字で再表示される。そして，ネットワーク上の次のデータが赤文字で表示され，そのデータに対する実行可能な関数が青枠で囲まれる。

【０１１８】このオペレーションウインドウで指定されたオペレーションに従って，データウインドウ（図示省略）へのオブジェクトの表示，消去，座標データの入力などの処理が行われる。

【０１１９】

【発明の効果】以上説明したように，本発明によれば，画像システムなどの特定応用分野のシステムをオブジェクトネットワークとしてモデル化することによって，容易に構築できるようになり，システム開発者にとっては，ソフトウェアシステムの生産性が向上するという効果がある。また，システムの状態や実行可能なオペレーションについて，わかりやすい形でユーザに提示できるので，ユーザにとっては，ユーザ・フレンドリなシステムの利用が可能であり，操作性が向上するという効果がある。

【図面の簡単な説明】

【図１】本発明の原理説明図である。

【図２】本発明に係るオブジェクトネットワークの参照モデルの例を示す図である。

【図３】本発明に係る名詞オブジェクト管理機構の構成例を示す図である。

【図４】本発明に係る動詞オブジェクトの動詞の種類を示す図である。

【図５】本発明に係る総称的関数と具体的関数の関係説明図である。

【図６】本発明の実施例に係る動詞オブジェクトの実行管理説明図である。

【図７】本発明に係る並行プロセス構築の層構造の例を示す図である。

【図８】本発明の実施例におけるトークンの制御方式を示す図である。

【図９】本発明の実施例による要素オブジェクトを生成する例を示す図である。

【図１０】本発明の実施例による画像描画用オブジェクトネットワークの例を示す図である。

【図１１】本発明の実施例によるトランスレータのフローチャートを示す図である。

【図１２】本発明の実施例に係る実行システムの構造を示す図である。

【図１３】本発明の実施例に係る実行システムのフローチャートを示す図である。

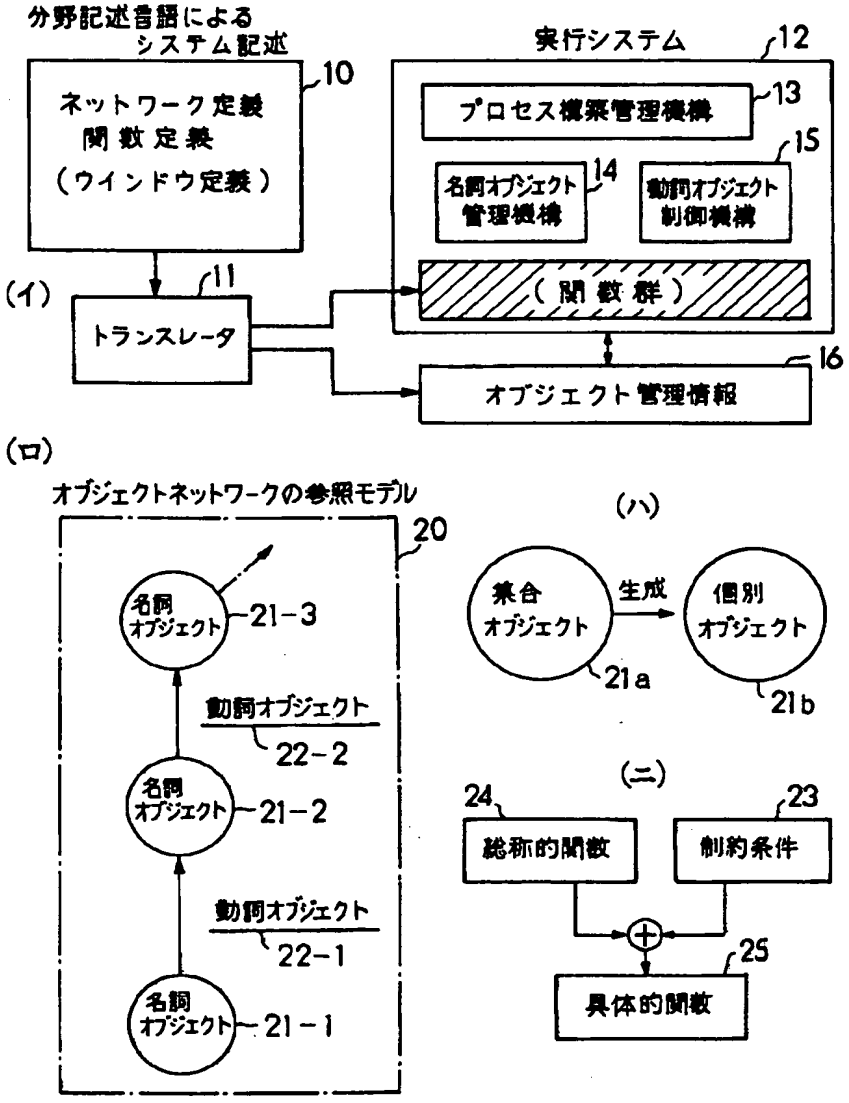
【図１４】本発明の実施例におけるオペレーションウインドウの例を示す図である。

【符号の説明】

１０　分野記述言語によるシステム記述
１１　トランスレータ
１２　実行システム
１３　プロセス構築管理機構
１４　名詞オブジェクト管理機構
１５　動詞オブジェクト制御機構
１６　オブジェクト管理情報
２０　オブジェクトネットワーク
２１　名詞オブジェクト
２２　動詞オブジェクト
２３　制約条件
２４　総称的関数
２５　具体的関数

【図1】

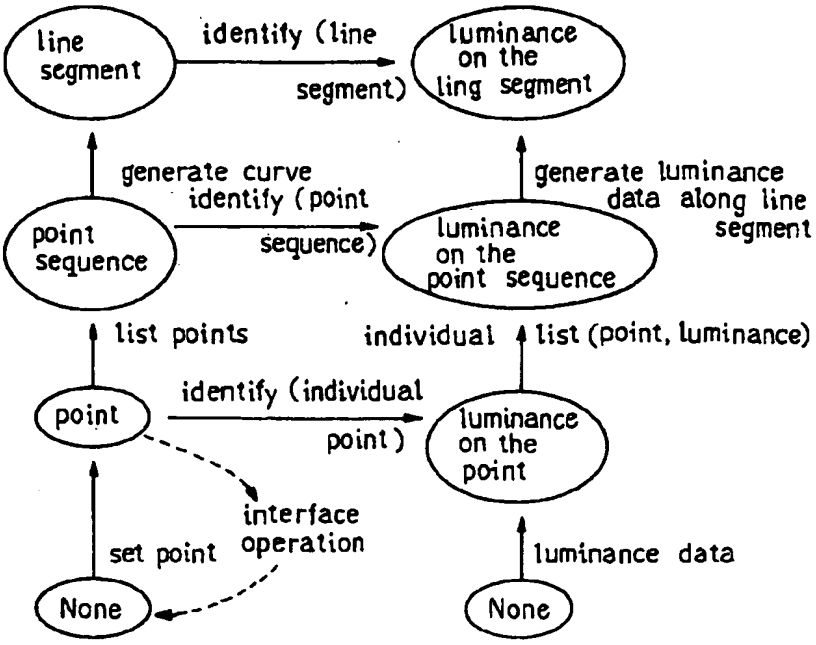
本発明の原理説明図



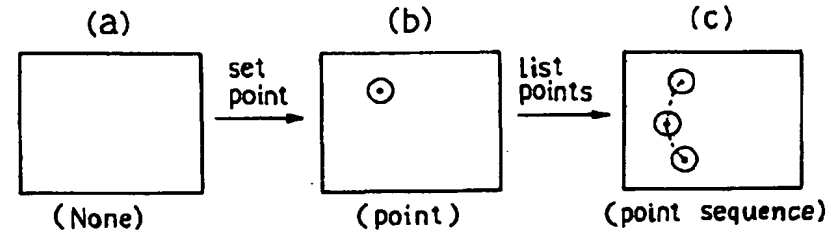
【図2】

オブジェクトネットワークの参照モデルの例

(イ)

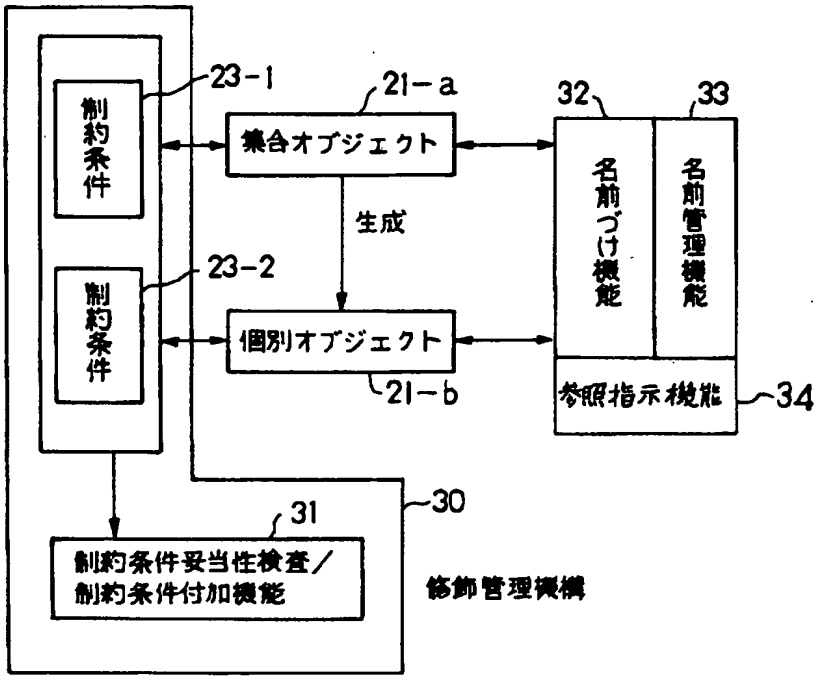


(ロ)



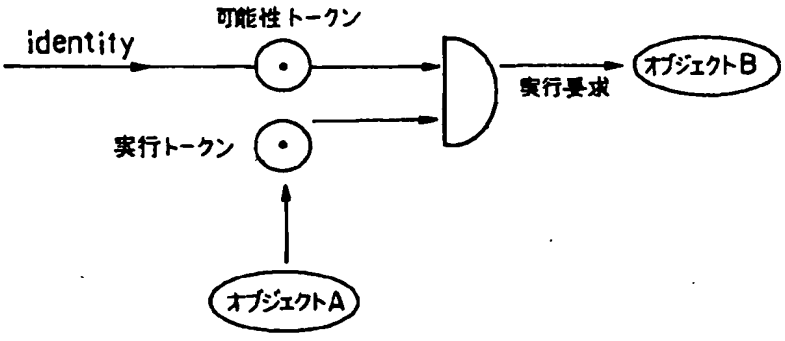
【図3】

名詞オブジェクト管理機構の構成例



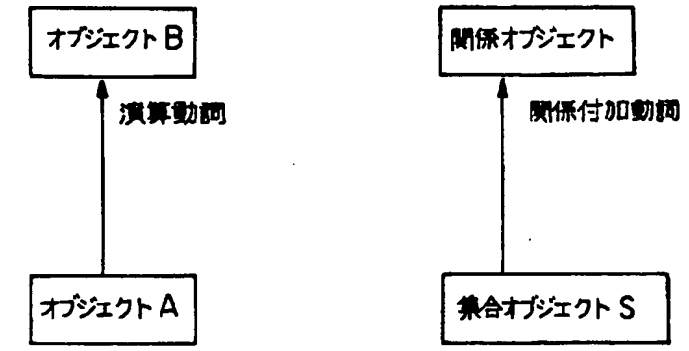
【図8】

トークンの制御方式

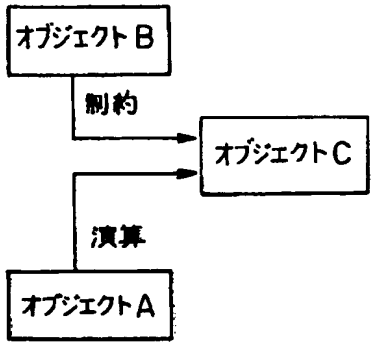


【図4】

動 詞 の 種 類



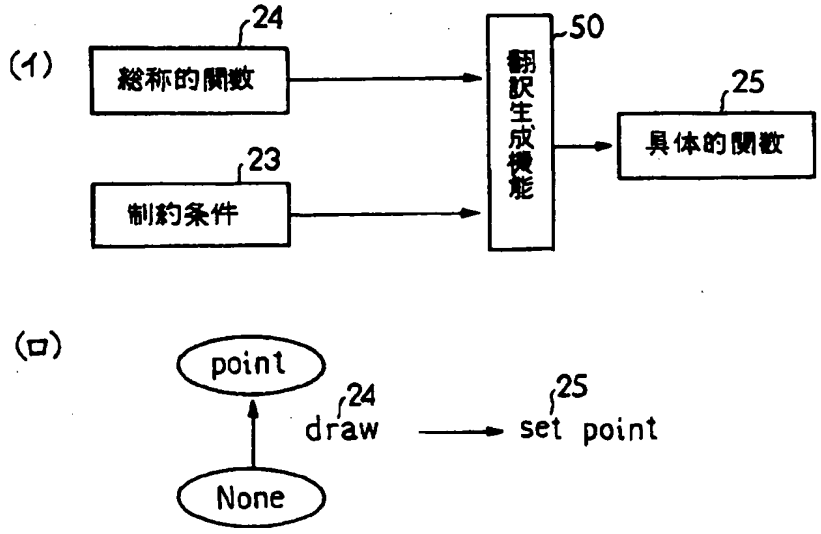
(a) 演算動詞　　(b) 関係付加動詞



(c) 制約演算動詞

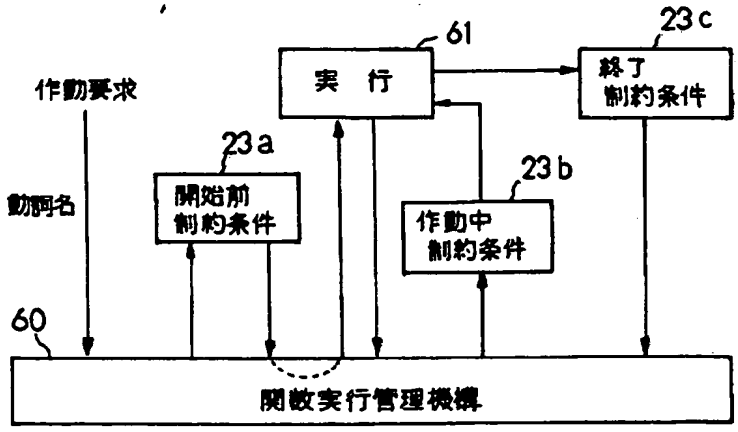
【図5】

総称的関数と具体的関数の関係

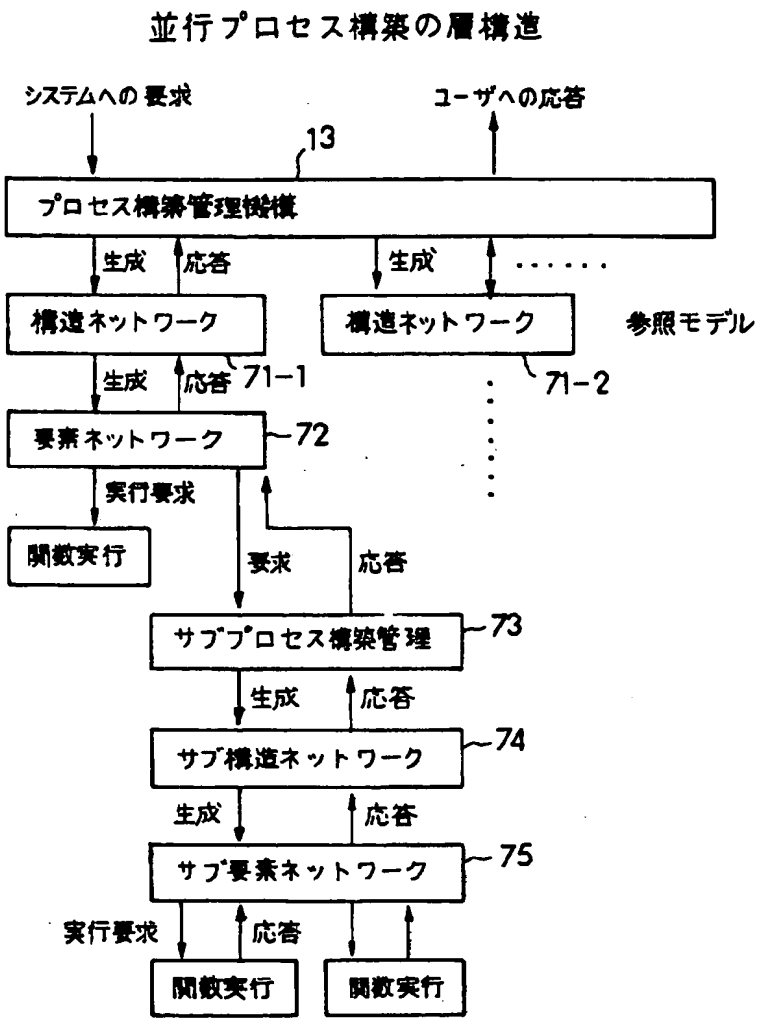


【図6】

動詞オブジェクトの実行管理

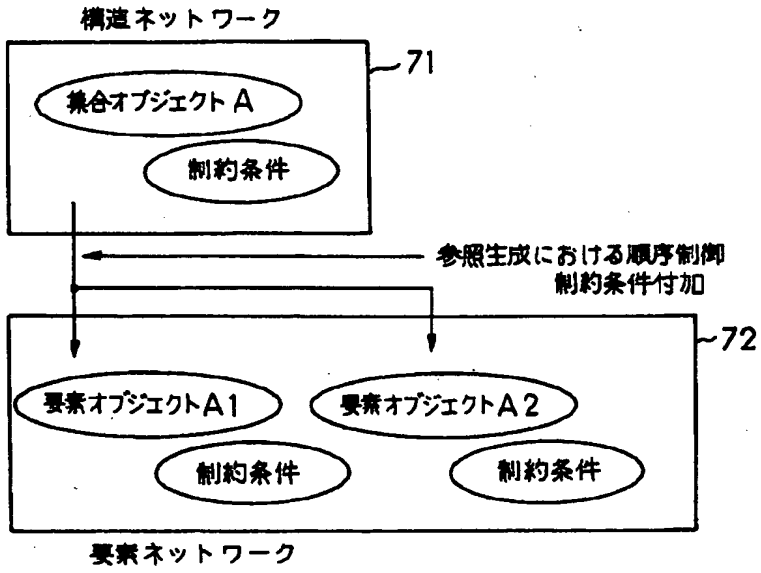


【図7】


並行プロセス構築の層構造
システムへの要求
ユーザへの応答
13
プロセス構築管理機構
生成
応答
生成
構造ネットワーク
構造ネットワーク
参照モデル
生成
応答
71-1
71-2
要素ネットワーク
72
実行要求
関数実行
要求
応答
サブプロセス構築管理
73
生成
応答
サブ構造ネットワーク
74
生成
応答
サブ要素ネットワーク
75
実行要求
応答
関数実行
関数実行

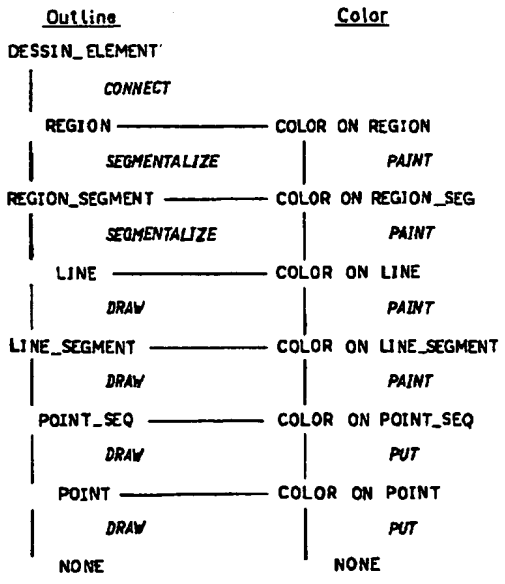
【図9】

要素オブジェクトの生成



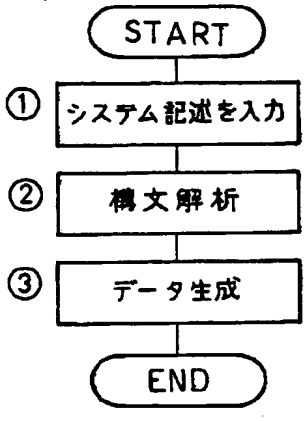
【図10】

画像描画用オブジェクトネットワークの例



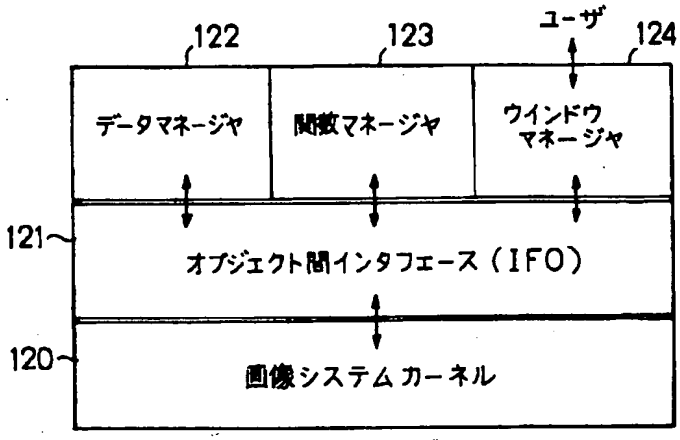
【図11】

トランスレータのフローチャート



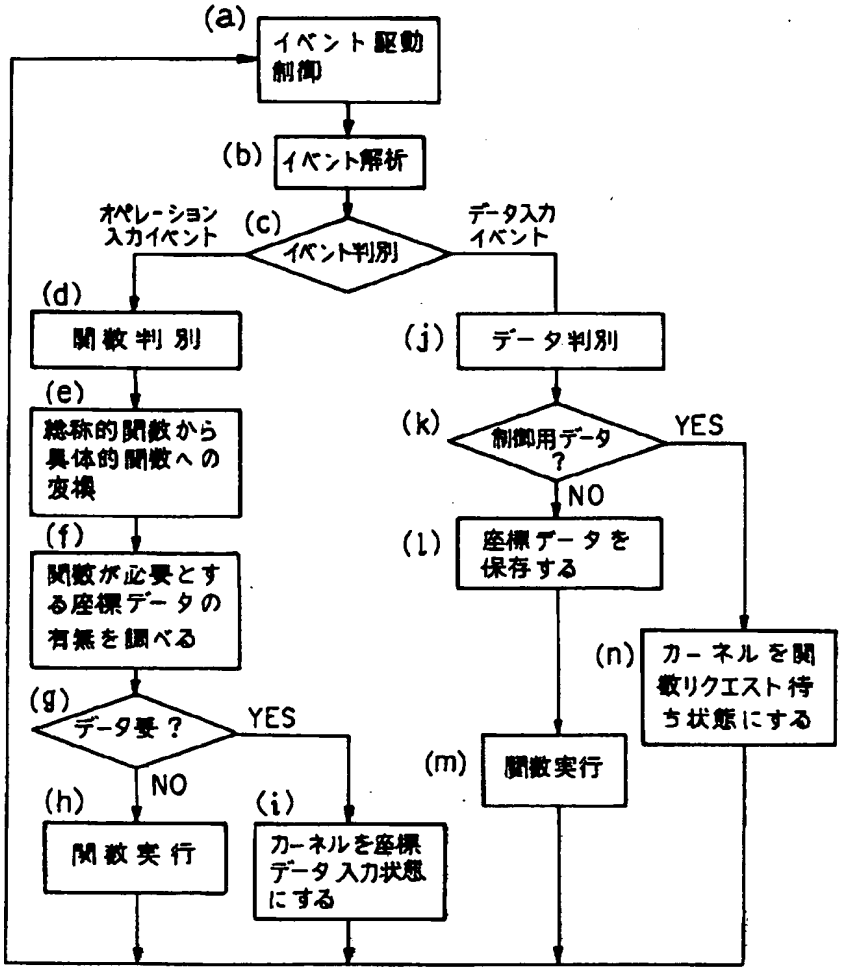
【図１２】

実行システムの構造



【図１３】

実行システムのフローチャート



【図14】

オペレーション ウインドウの例